PREFACE

This Paper is essentially a progress report on the simultaneous use of on-line, time-shared JOSS* computer consoles at The RAND Corporation. It covers activities implemented during the past six months by this writer and other members of the RAND staff. It is a preliminary report in the sense that large scale applications of the concepts presented herein will take place in the near future, but extensive gaming results are not yet available. It is a progress report in the sense that most of the groundwork (programming) has been completed and tested and it is possible to state categorically that the basic concepts work. Man-machine gaming situations and simulations have been successfully accomplished.

Throughout this Paper the use of The RAND Corporation's JOSS System in on-line, time-shared multiconsole gaming and simulation is described in considerable detail. Naturally, other on-line, time-shared computer systems are commercially available. In some cases they may be superior to the JOSS System. Unfortunately, time has not permitted a comparison of the advantageous features of different computer systems. However, the reader should bear in mind that neither the JOSS System nor the techniques described herein are being extolled as the best or preferable way to accomplish the task of bringing closer together computer power and the human participants in games or simulations. To reiterate the opening statement, this Paper is a progress report of what has been accomplished. It is left to the reader to conjecture how far we can go in the applications of on-line, time-shared computer consoles in games and simulations, and how advantageous this will be.

---

*JOSS is the trademark and service mark of The RAND Corporation for its computer program and services using that program.

ABSTRACT

Some present-day on-line, time-shared, multiple-console computer systems provide for use of a common file system. One console can file a message (i.e., "information") which can be recalled by another console. By programming consoles to periodically interrogate certain files, a crude, but highly serviceable, store-and-forward communication system can be created and large numbers of on-line, time-shared computer consoles can be used to enter, recall, process, and display information typical of that used in command and control systems and the play of games.

The RAND Corporation's JOSS System provides the capability described. In addition to its use for the solution of scientific problems, it is presently being employed to simulate in real time elements of an automated tactical air control system and in the play of tactical games and games of global strategy. The simple, easy-to-learn programming language makes feasible considerable experimentation with scheduling algorithms, decision rules, etc.

This Paper describes the basic features of the use of multiple JOSS consoles in simulation and gaming and discusses some of the advantages, limitations, and lessons learned to date.

ACKNOWLEDGMENTS

This Paper has benefited from the work and constructive suggestions of C. F. Black, R. L. Clark, D. C. Kephart, and E. W. Paxson. All errors, of course, remain the responsibility of the author.

USE OF MULTIPLE ON-LINE, TIME-SHARED COMPUTER CONSOLES IN SIMULATION AND GAMING

G. M. Northrop*

The RAND Corporation, Santa Monica, California

I. INTRODUCTION

The need for data automation as an aid to real-time or near real-time system simulation and gaming is a long-standing one. Past efforts have seen many applications of digital computers in this area.(1,2) In general, however, there has been an inherent difficulty in applying first and second generation computers to real-time simulation and games of strategy involving both men and machines operating in real-time. The difficulties in applying digital computers in the past derived in part from the lack of computational speed, the complexities of programming languages, and the batch operation feature (single input, single output) of most computers of that period.

To perform simulations and gaming in real-time or near real-time, it has long been recognized that all participants, including the control group, have need for continuously available computer power. The system should also be capable of performing the functions of a store-and-forward communications system so that actions taken by one party can--at the direction of the initiator or as an inherent function of the nature of the action--provide raw and/or processed information to all other affected parties in the game or simulation. The computer power provided to the participants needs to be of sufficient speed and

---

*Any views expressed in this paper are those of the author. They should not be interpreted as reflecting the views of The RAND Corporation or the official opinion or policy of any of its governmental or private research sponsors. Papers are reproduced by The RAND Corporation as a courtesy to members of its staff.

This paper was prepared for presentation at the National Gaming Council Symposium, Washington, D.C., June 8, 9, 1967.

storage capacity so that each participant in essence can employ programs that provide him with information or capability at least approximately equivalent to computer equipment and/or data processing and communications and display equipment that might exist in an actual situation.

Today's third generation digital computers, programmed and configured to be used in an on-line, time-shared fashion, can meet or closely approximate the general requirements outlined above. Obviously, the many various equipment arrangements and programming languages that are presently available provide many degrees of capabilities in each of the simulation and gaming needs suggested.

This paper describes briefly some of the characteristics of The RAND Corporation's JOSS System* -- an on-line, time-shared computer presently providing service through 34 consoles. The basic characteristics of the JOSS System storage file configuration make it possible for one console to command the entering of information, either textual, data, or both, into a file-item that can be accessed by programs under the control of any other console in the system. This feature permits the use of the JOSS System as a crude, but serviceable, store-and-forward communication system, in addition to its usual role of data processing.

Three recently developed applications of the JOSS System to multiconsole** on-line simulation and gaming are discussed herein. The first application presented puts the JOSS System in the role of message handler for a BLUE-RED game under control of GREEN (the referee). Although numerical data processing does not take place in this game, it can easily be added in the future. A two-party battle between missile-firing submarines, in which extensive data processing occurs, constitutes the second application. The third application is a simulation of some facets of the close air support role of a hypothetical automated tactical air control system.

---

*JOSS is the trademark and service mark of The RAND Corporation for its computer program and services using that program.

**Multiconsole operation implies many game or simulation participants, each operating a console and programs in real-time.

The role played by a modern multiconsole, on-line, time-shared computer in simulation and gaming is by no means limited to the illustrative applications given in this paper. As noted above, necessary ingredients tend to be large computer power,* mutual accessibility of files by all consoles, and a convenient, easy-to-use programming language. While all of these features are important, the last is probably most nearly paramount, for it is the feature that permits the real users of the system to improve and improvise rapidly and easily as the development of the game or simulation progresses. Fortunately, these three basic system elements are available today. And it is probably safe to say that future applications of on-line, time-shared computers in games and simulations will be widely varied, extensive in scope, and very imaginative.

---

*"Computer power" is considered here to be large when 1) the computer has the capability to perform the basic add-subtract operation in no more than a few microseconds; 2) there is "sufficient" high speed core storage; and 3) there is "adequate" direct access storage.

II. THE JOSS SYSTEM

RAND's JOSS System was first implemented on the JOHNNIAC computer (now residing in the Los Angeles County Museum) by J. C. Shaw, to whom principal credit must go, both for design and construction. Beginning with partial operation in early 1963 and progressing to full capability with 8 on-line consoles in January 1964, the present JOSS System has been succinctly described by G. E. Bryan in a recent paper:[3]

> JOSS is a time-shared computer system that provides for the solution of numerical problems via an easily learned language at remote typewriter consoles. The PDP-6 hardware used to implement JOSS consists of 32,000 words of 1.75μsec core memory, a 1-million-word 4μsec drum, a 6-million-word discfile, and various peripheral devices. A special data relocation mode for memory references has been added to facilitate interpretation of JOSS programs. The JOSS consoles, built around a Selectric I/O typewriter, were specially manufactured to RAND specifications. Features include full duplex signaling, line parity checking, a page ejection mechanism, and several buttons and lights to control and report console status. The stand-alone JOSS software consists of the JOSS language interpreter and its arithmetic subroutines, a monitor for user scheduling and resource allocation, and I/O routines for the disc, drum, consoles, and other peripheral devices. JOSS service is currently available to nearly 500 users through 34 consoles, six of which are remote to RAND operating over both private and dataphone lines.

For the purposes of real-time gaming and simulation, the JOSS language is sufficiently close to conventional English to permit the user to begin writing some parts of the programs needed for simulation and gaming after only a few minutes of instructions. The JOSS System software provides a high degree of interaction (i.e., feedback) with the programmer during the course of writing the program, facilitating on-line programming, and program checkout and debugging. Each command is tested by the JOSS System for certain programming requirements, and it is also possible to ask JOSS (JOSS users often refer to the System collectively by the personal pronoun) to execute each line or a given group of lines of instruction to test for validity.

In the Fall of 1966, JOSS System software designers added a feature that makes possible the use of many consoles in a simultaneously interactive mode, and, thus, makes JOSS suitable for multiconsole games and

simulations. The added feature was the ability to do filing, recalling, and discarding of information under program control. Since all consoles share a common disc file, this feature carries with it implicitly the ability for all consoles to jointly interact with the same file. They must do so, of course, sequentially. At present, the techniques for sequencing and timing file actions needed to avoid conflicting file usage are left to the programmer (the user of JOSS). In the near future it is expected that more appropriate features will be added to the JOSS System software and will become available to all JOSS users.

A typical example of the use of a JOSS file and the repeated interrogation of a JOSS file-item is shown in Fig. 1. This figure also gives a good illustration of the ease with which JOSS programs can be written by even the newly initiated. The flow diagram for the program (part 1, in JOSS language) of Fig. 1 is shown in Fig. 2. With the help of the flow diagram it should be apparent that the program permits the user to enter the file and then repeatedly interrogate* the file-item (item 5 (MSAGE)) where it is anticipated that some other user will have placed a message.

The message is assumed to be composed possibly of data, or text and data, or text, in conjunction with an identifying parameter, "k," whose numerical value signifies the message type. For example, k = 0 means no message present, k = 1 means data only, k = 2 means text plus data, and k = 3 signifies that only text is present. The text is assumed to be written as "part 10," thus it can be passed around through the system under that general designation and typed out (displayed) at any desired point. For the cases where the message includes data, the designated program (part 2 or part 3) would process the data and display results for the message recipient. As shown in Fig. 2, part 2 or part 3 would return to message monitoring. However, other options are also possible.

---

*Ultimately, it is to be expected that JOSS will become a full and equal gaming partner, signaling other consoles, making his own decisions, and having his own suspense file.

```
8:30  5/9/67    #13  GMN  1423      [10]

Type all.

1.1 Use file 234 (syop4).
1.2 Recall item 5 (MSAGE).
1.21 To step 1.7 if k = 0.
1.3 Do part 2 if k = 1.
1.4 Do part 3 if k = 2.
1.5 Type part 10 if k = 3.
1.7 Do step 1.72 for n = 1(1)2000.
1.71 To step 1.8.
1.72 Set n = n.
1.8 To step 1.2 if k = 0.
1.81 Set k = 0.
1.82 Discard item 5 (MSAGE).
1.83 File k as item 5 (MSAGE).
1.9 To step 1.2.
```

Fig. 1--Typical JOSS program for message interrogation, decision, and time delay.

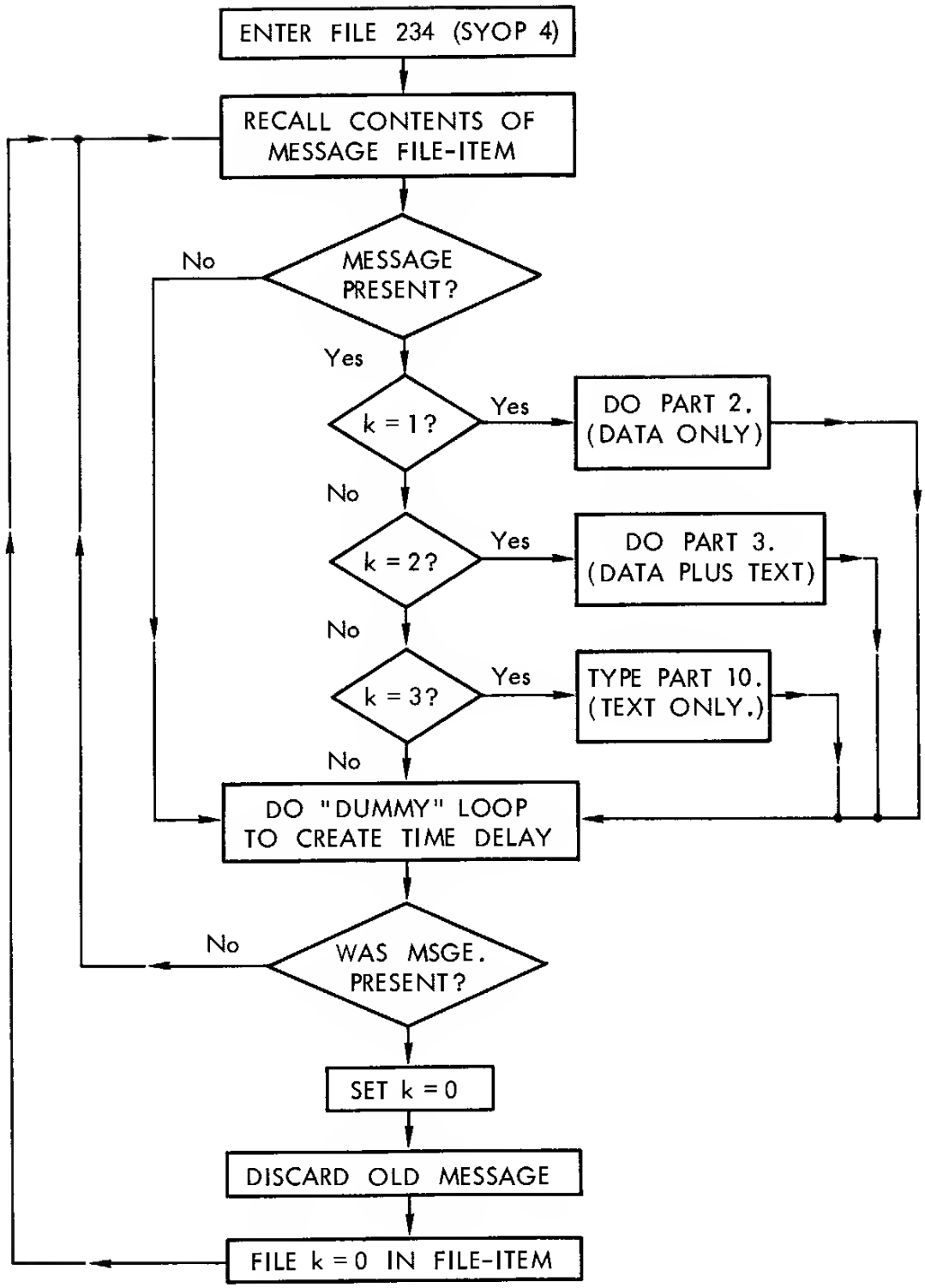


Fig.2—Flow diagram of typical interrogation, decision, and time delay JOSS program

The "dummy" Do Loop is used to create a time delay between interrogations of the message file-item. This is a brute-force technique for creating delay between looks; a more satisfactory solution will be added to the JOSS System software in the near future.

After a message has been received and appropriately processed and displayed, the file-item is emptied and a value, k = 0, filed in it to indicate no message is present, and the program reverts to cyclic interrogation. A new message can be entered at any time by a game player at another console, who replaces the contents of the file-item with the new message and a message-type designator value for the parameter, k.

III. MULTICONSOLES IN A TEXT-ONLY WAR GAME

The file-item interrogation program of Fig. 1 can be combined with a message generation program and a start-up routine to provide the essential features of a store-and-forward communication system suitable for passing messages back and forth between two combatants in a war game in which, for example, diplomatic exchanges are the principal feature. A schematic outline of a simple two-player configuration is shown in Fig. 3. The use of the start-up routine is shown in Fig. 4 and the salient features of the message generation program are illustrated in Fig. 5.* The three simple programs needed to provide this level of gaming capability are given in Appendix A.

For the more interesting game applications, the exchange of messages between BLUE and RED would be monitored and influenced by the referee, GREEN. The basic start-up, message generating, and interrogation program of Appendix A can be modified somewhat to permit BLUE-RED messages to be monitored simultaneously by GREEN, and to afford GREEN the ability to send private messages to BLUE or GREEN.** The fundamental structure of such a console configuration is shown in Fig. 6.

In this instance, six file-items have been used as intermediate storage points for messages from the participants. This ensures, for example, that BLUE action in response to a message from RED will not prevent GREEN from getting a copy also. As indicated in the figure, if desired, GREEN can simultaneously operate one computer console in the message receiving mode and a second in the message transmitting mode, thus assuring up-to-the-minute knowledge of all communications between BLUE and RED, without slowing preparation and transmission of GREEN messages.

---

*Console user inputs appear in green type; JOSS responses appear in black type. This feature of the JOSS System is used throughout this Paper.

**Because the operations performed by GREEN are slightly different from those executed by BLUE or RED, GREEN uses slightly modified versions of the start-up, message generating, and interrogation programs that are jointly used by BLUE and RED. These six programs are given in Appendix B.

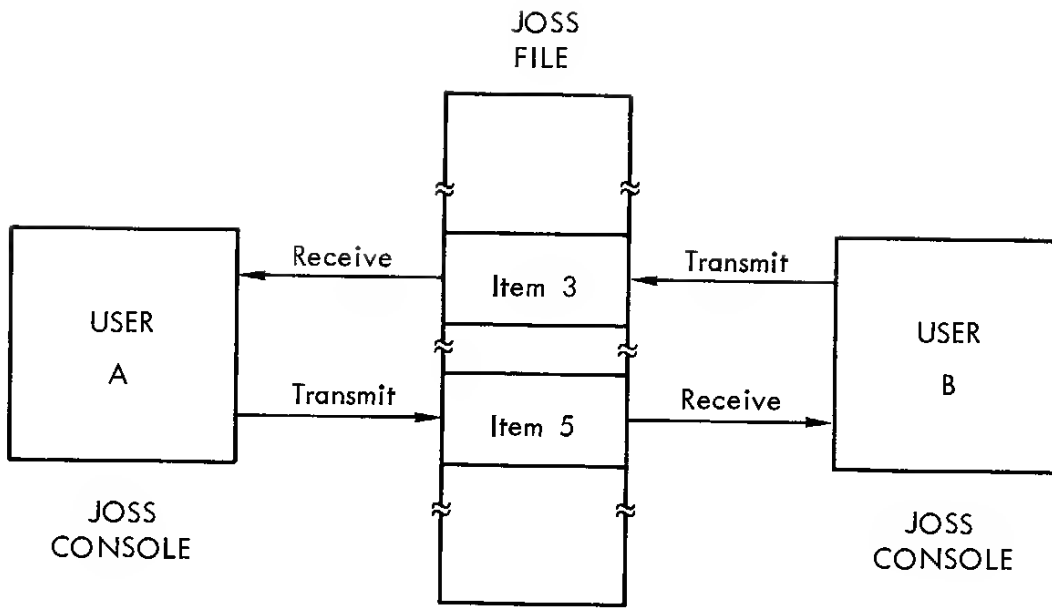


Fig. 3—Typical two-party JOSS store-and-forward communication system configuration

```
Use file 233 (syop3).
Roger.
Recall item 1 (COMM1).
Done.
Do part 10.
JOSS Store-and-Forward COM-SYSTEM Start-up Program.


Give item no. to which your messages are to be transmitted.
        T = 3

Give item no. from which you will receive messages.
           R = 5

To begin operation in the RECEIVE mode, type K = '0'.
To begin operation in the TRANSMIT mode, type K = '1'.
           K = 1

END OF START-UP PROGRAM.
```

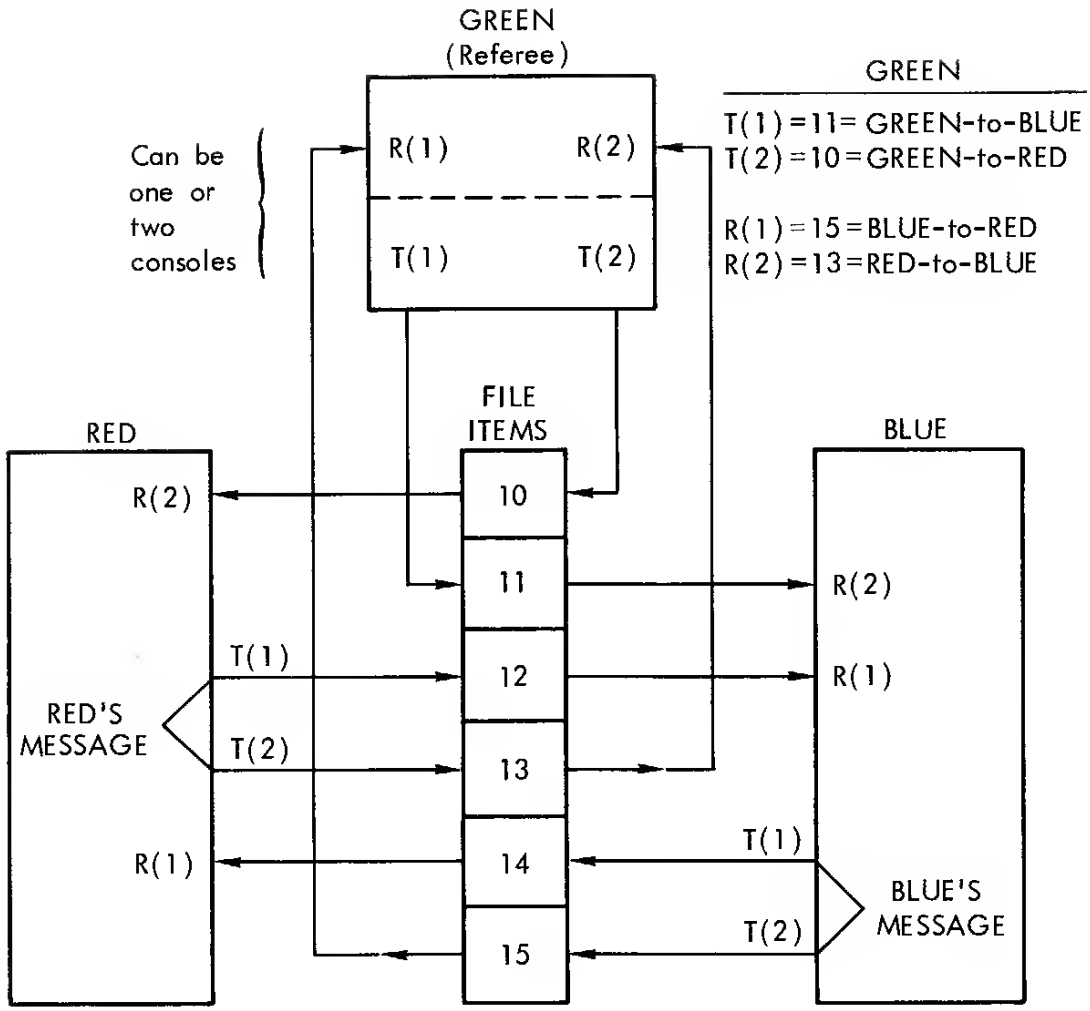
Fig.4—Use of JOSS start-up program

```
MESSAGE GENERATION AND TRANSMISSION PROGRAM.


Preface each line of message with a "Part 1" number.
After composing message, type "Go.".
Stopped by step 100.4.
1.1 TIME = 11:42 hrs
1.11
1.2 FIRST MESSAGE FROM RED TO BLUE:
1.21
1.3 ####  BLUE military actions in the vicinity of HUMDRUM PASS
1.31      represent a violation of our borders. RED military forces
1.32      are retaliating against #LUE#airbases and military installations
1.33      within 200 n mi of HUMDRUM PASS.
1.4
1.41      His Royal Highness of RED-LAND has requested an immediate
1.42      session of the UN SECURITY COUNCIL.
1.5
1.51                SIGNED:   First Premier of RED-LAND
Go.
1.1 TIME = 11:42 hrs
1.11
1.2 FIRST MESSAGE FROM RED TO BLUE:
1.21
1.3       BLUE military actions in the vicinity of HUMDRUM PASS
1.31      represent a violation of our borders. RED military forces
1.32      are retaliating against BLUE airbases and military installations
1.33      within 200 n mi of HUMDRUM PASS.
1.4
1.41      His Royal Highness of RED-LAND has requested an immediate
1.42      session of the UN SECURITY COUNCIL.
1.5
1.51                SIGNED:   First Premier of RED-LAND


To make CORRECTIONS, set C = '1', else set C = '0'.
          C = 0
MESSAGE TRANSMITTED.
To shift to MESSAGE MONITORING mode, set k = '1'.
To send ANOTHER MESSAGE, set K = '0'.
          K = 1
Entering the MESSAGE MONITORING mode.
```

Fig.5—Use of JOSS message generation and transmission program



RED

T(1)=12=RED-to-BLUE
T(2)=13=RED-to-BLUE (to GREEN)

R(1)=14=BLUE-to-RED
R(2)=10=GREEN-to-BLUE

BLUE

T(1)=14=BLUE-to-RED
T(2)=15=BLUE-to-RED (to GREEN)

R(1)=12=RED-to-BLUE
R(2)=11=GREEN-to-BLUE

Fig.6—Typical BLUE-RED-GREEN wargame JOSS configuration

The natural extensions of the two brief examples to configurations involving separate send and receive consoles at BLUE, RED, and GREEN locations is straightforward and can be accomplished with the same programs that would be used if there were but one console at each location. For cases where game data, rather than just textual messages, is to be transmitted, computational algorithm programs can be added to show, for example, the effect of expenditure of resources by one combatant against the other.

In its role as referee, GREEN could monitor such exchanges of data, supplying to the expender of resources certain parameter ranges and/or distributions, so the expender of resources could use the computational algorithms to test the range of effectiveness of the proposed action before actually committing the resources. Then, when the expenditure finally takes place, GREEN would supply the recipient of the attack with the actual parameter values to be used (automatically, within the algorithm program) in determining the effectiveness of the attack. In instances where greater automaticity of play is desired, the ranges and/or values of parameters could be set by a random number generator provided within GREEN's programs.

It is evident that once the basic programs for the exchange of information have been made operable--as indicated by these examples--they can be easily expanded to include many of the more desirable and esoteric features of game operation. One of the more entrancing features is that once the store-and-forward message handling feature has been provided, the builders of the game can add increasingly sophisticated computational features to the total program structure and perform checkout tests of new additions in an on-line, real-time mode of operation. Another desirable feature is the ability to "stop the clock" and allow the players to perform "what if?" calculations, e.g., "What are the expected results, if I decide to execute Option X at this point of game play?"

IV. MULTICONSOLES IN A SUB-LAUNCHED MISSILE DUEL

To develop background information for the use of the JOSS System in war gaming at RAND, C. F. Black and D. C. Kephart have structured programs for a simulated battle between two missile firing submarines--extending the two-console "JOSS-SPIEL" game initially formulated by T. A. Brown. This is a two-player game, with each player given a trade-off choice of four combinations of maximum sub speed and a number and range of missiles. A player does not know initially what capabilities his opponent has chosen, but as the game progresses it may be possible to infer this.

The two players operate within a hypothetical 100-by-100 n mi square of ocean. The hypothetical sub-launched short range missiles are assumed to have a kill radius of 2 n mi and a maximum range of 60 or 75 n mi, depending on the model. Each player also positions 4 nuclear mine fields (4 n mi damage radius) in the square. The players then input the position, heading and speed of their submarines and the game begins. A list of game rules is given in Fig. 7.

The program carries the play through in assumed 15-minute time intervals of action. Random number generators are used to create uncertainty in knowledge of the location of the opponent and in the reliability of missile launch.

To keep the operation of the two consoles in proper sequence, a trial "flag" is set to zero at the beginning of the game and incremented by the opponents as play progresses. The value of the flags must be equal before play can move to the next stage. Figure 8 shows a schematic of the submarine game operation.

The play is continued at a reasonable pace by using the timer provided as part of the JOSS System software and allowing each player only two minutes to choose a new course heading, new speed, and fire up to three missiles at selected aim points.

The two players make their moves essentially simultaneously. After both players have specified their actions for the next 15-minute interval, the various parts of the program are activated and the results determined and presented to the players. Since missiles fired at time

- Missile kill radius = 2 n mi.
- Damage radius = 4 n mi (missiles and minefields).
- Damage: Yaur speed and maneuver contrals are lacked at present values. Fire cantral stays aperable.
- Damage + damoge = kill.
- Yau get 2 minutes real time ta maneuver yaur baat and launch missiles.
- Salva limit = 3 missiles.
- Missile CEP = 0.
- Missile range = 60 n mi (except mad 1 = 75 n mi).
- Accuracy af sensor fix an enemy:
  Range: ± 10 percent as perturbed by a random number.
  East and Narth caardinates: ± 10 percent af range. (Each caardinate independently perturbed by a different random number).
- 50 percent speed penalty ta execute o caurse reversal.
  25 percent penalty far 90-deg turn. Others in prapartian.
- Each time intervol covers 15 minutes af aperatian.
- Zera speed for 15 minutes if yau run aground while undamaged.
- Yau can slip thraugh a minefield—as lang as yau are nat still inside the minefield at the end af a 15-min step interval. You can be clabbered twice in the same minefield if yaur speed is slow enaugh.
- Missiles targeted ta impact paints beyand ronge are last due ta cammander's grass errar.
- Yaur cammitments are final ance yau hit the JOSS "RETURN" key.
- Yau get na damage repart an the enemy. Yau might deduce his status by abserving his maneuvers.
- Missile launches shauld be targetted against expected enemy lacatian 15 minutes after launch.

Fig.7—Rules for the two-player hunter-killer submarine game

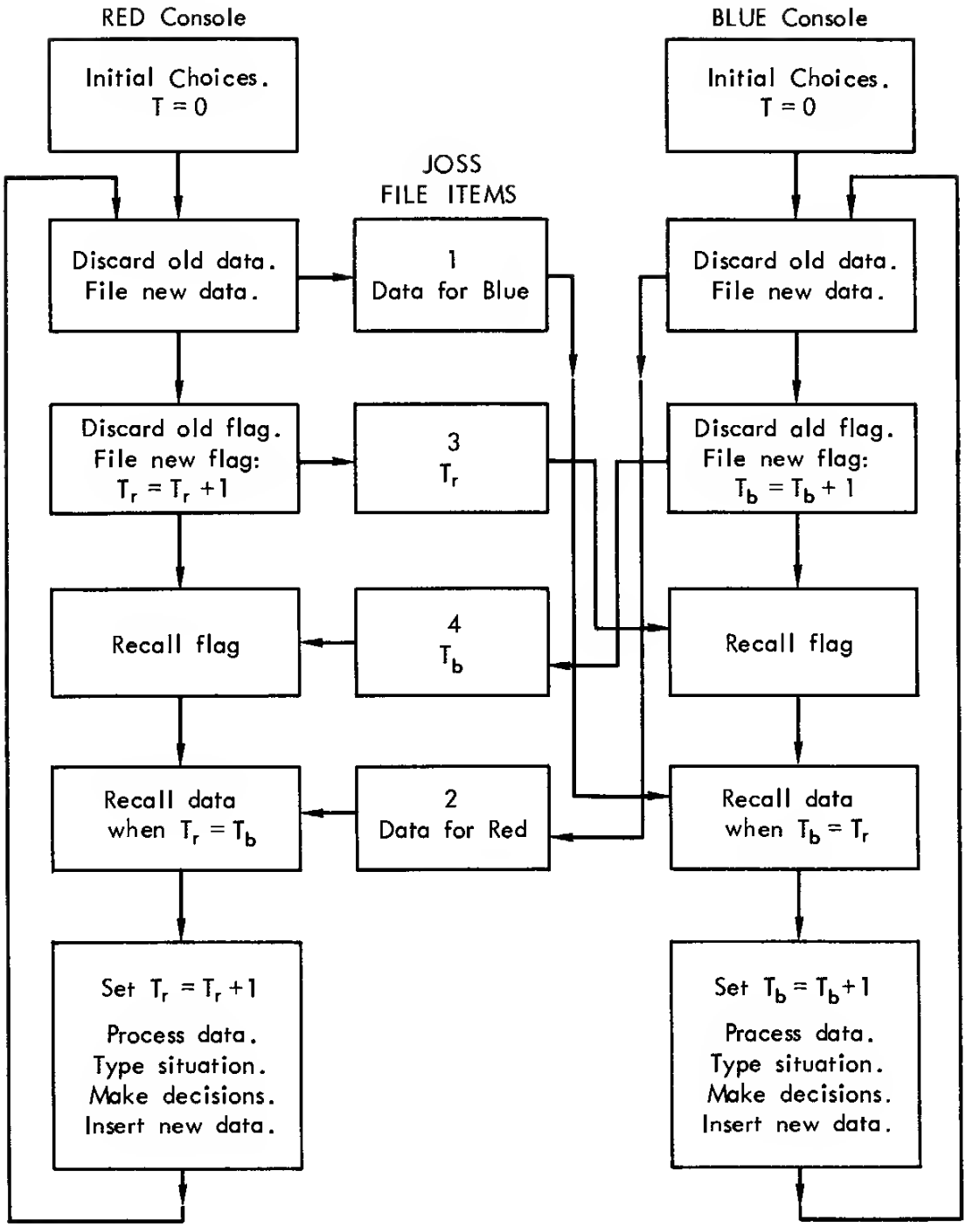


Fig. 8—JOSS submarine battle simulation: data channels and action sequencing

step T do not impact until T + 1 (a 15-minute game cycle later), then each submarine has an opportunity to avoid both enemy and own shots, and players must make a "lead" allowance in targeting the opponent. Figure 9 shows an initiation of game play, as seen by BLUE.

Because of the flexibility of the JOSS language, it has been possible to make the programs very general: with little additional effort, new decision rules and weapon systems, such as torpedoes, depth charges, active and passive acoustic sensors, ocean depth constraints, and communications links (including links between adversaries) could be added. With additions such as these, this and similarly structured games (tank battles, surface ship conflicts, etc.) could be useful both in developing tactics and as a training aid for actual weapon system users.

```
Use file 76 (2890).
I can't find the required file.
Use file 76 (K2890).
Roger.
Recall item 7 (CAME7).
Done.

Do part 10.
 :: :: HUNTER KILLER NUCLEAR SUBMARINE ENCOUNTER :: ::

RED: i = 1   BLUE: i = 2
          i = 2

You may choose your Weapon System:
L=2  Max speed = 26 kts. You get 13 missiles.
L=3  Max speed = 33 kts. You get 11 missiles.
L=4  Max speed = 43 kts. You get  9 missiles.
L=5  Max speed = 53 kts. You get  7 missiles (Mod 1).
Missile range = 60, Reliab = .90. Mod 1: Range = /5, R=1.0.
       L(2) = 5
Give initial position: Miles East and North of (0,0).
          E = 70
          N = 50
Order initial SPEED (s) and COURSE (c):
          s = 53
          c = 315
You may plant 4 MINE-FIELDS.  Say where:
          E = 70
          N = 70

          E = 50
          N = 60

          E = 30
          N = 50

          E = 50
          N = 40

  0  Time:1200 Your position:  70 East,    50 North.
                  Speed:  53 kts,  Course: 315 deg.
```

Fig.9—Hunter-killer submarine game: BLUE printout

```
  1  Time:1215 Your position:  61 East,    59 North.
                  Speed:  53 kts,  Course: 315 deg.
::
  ::::  FLASH  ::::
ENEMY CONTACT *** ENEMY CONTACT  ::: BATTLE STATIONS :::
          ENEMY RANGE  45 mi.  ENEMY POSITION  24 EAST,  32 NORTH
::
::

You have 2 minutes to maneuver and launch missiles.
Give COURSE Command:
          c = 0
Give SPEED Command:  (percent of max. speed)
          p = 100
You have   7 missiles.
Give aim coords, (E,N).   E = 111 if done.

          E = 23
          N = 31
50 sec to go.

          E = 111
Error at step 9.4:  I can't find the required item.
Go.
```

Fig.9(cont.)—Hunter-killer submarine game: BLUE printout

```
  2  Time:1230 Your position:  61 East,    71 North.
                    Speed:  46 kts,  Course:   0 deg.
::
           ENEMY RANGE  55 mi.  ENEMY POSITION  35 EAST,  23 NORTH
::
Thump
::
:: ENEMY ATTACKING ::
::: BLAM :::

You have 2 minutes to maneuver and launch missiles.
Give COURSE Command:
           c = 45
Give SPEED Command:  (percent of max. speed)
           p = 100
You have   6 missiles.
Give aim coords, (E,N).   E = 111 if done.

           E = 111




  3  Time:1245 Your position:  69 East,    79 North.
                    Speed:  46 kts,  Course:  45 deg.
EMERGENCY::::: We have Reactor Trouble.
Maximum Speed reduced to 41 knots.
::
           ENEMY RANGE  78 mi.  ENEMY POSITION  31 EAST,  16 NORTH
::
::

You have 2 minutes to maneuver and launch missiles.
Give COURSE Command:
           c = 0
Give SPEED Command:  (percent of max. speed)
           p = 0
You have   6 missiles.
Give aim coords, (E,N).   E = 111 if done.

           E = 111
Status:  RED ALERT.
Status:  RED ALERT.
Error at step 9.4:  I can't find the required item.
Go.
```

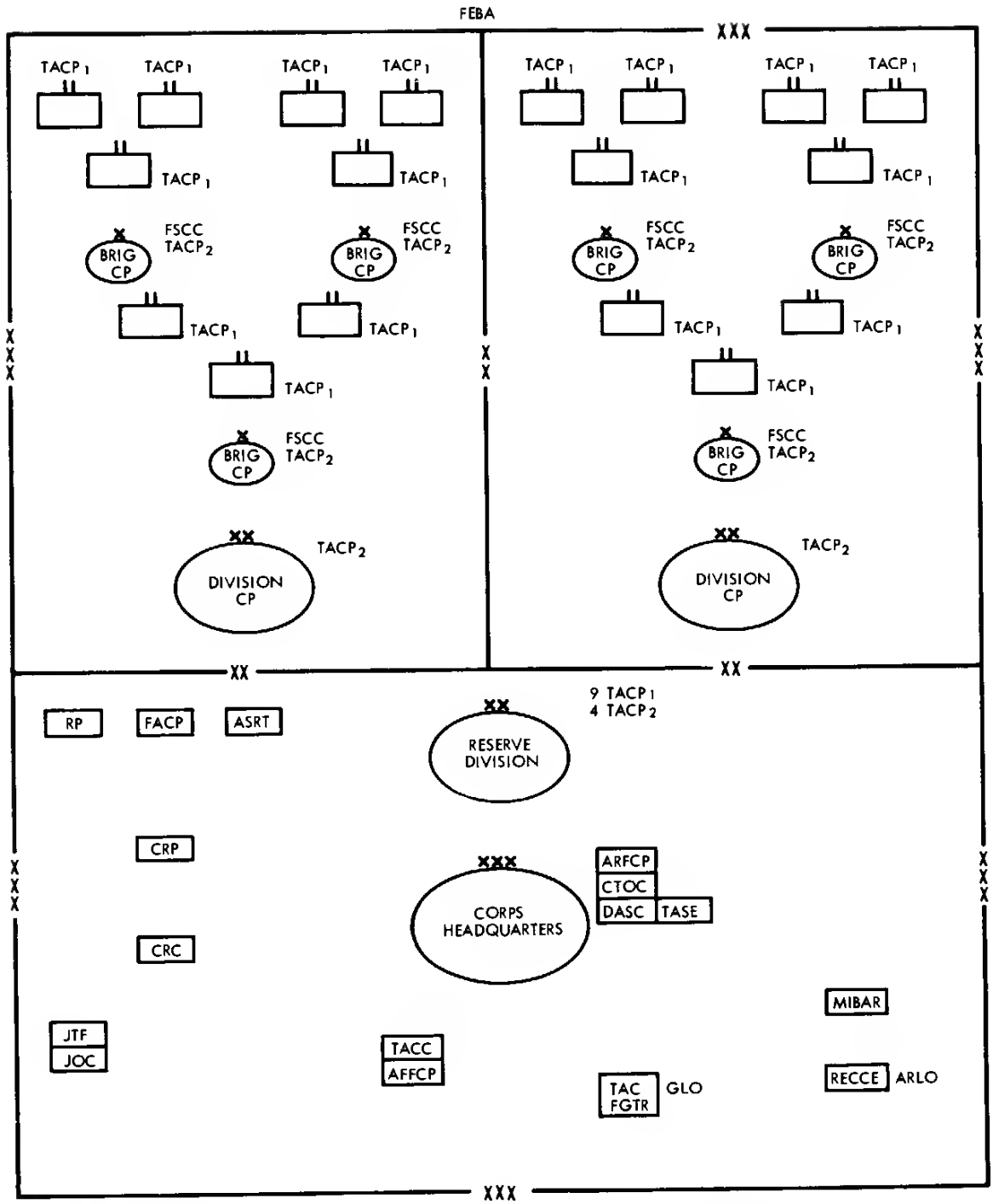
Fig. 9 (cont.)—Hunter-killer submarine game: BLUE printout

V. MULTICONSOLES IN THE SIMULATION OF AN AUTOMATED TACTICAL AIR CONTROL SYSTEM

In addition to uses such as those described in the previous two sections, the JOSS System has recently been used in a simulation developed by this author of a part of a hypothetical Automated Tactical Air Control System. In the present day U.S. Air Force Tactical Air Control System, the more urgent messages are usually transmitted in voice form and must be hand copied for further action. Less urgent messages may be transmitted by teletype, but this, too, may result in time delays before the information can be processed. Modern data automation could likely reduce these time delays, and in addition might provide such benefits as greater accuracy for certain staff actions requiring computations and/or sorting, searching, comparing, and correlating certain data prior to making basic decisions.[4]

COMMAND AND CONTROL IN THE PRESENT SYSTEM

The relationship of the present Tactical Air Control System and the U.S. Army units it supports is shown in Fig. 10 for a typical Army corps area of responsibility in a joint task force operation. The corps front, or Forward Edge of the Battle Area (FEBA), might be 20 to 60 miles in width. The corps consists of at least two divisions; in Fig. 10, two divisions are on the FEBA and a third is in reserve in the rear. Each division contains three brigades and each brigade, three battalions. The Air Force probably would provide air defense and air control elements such as the Control and Reporting Center (CRC), Control and Reporting Post (CRP), etc., as shown in the lower left of Fig. 10. Other Air Force elements would likely be the Tactical Air Control Center (TACC) associated with the Air Force Forces Command Post (AFFCP) and a Direct Air Support Center (DASC) operating in conjunction with the Army's Tactical Air Support Element (TASE). Other Army command elements include the Corps Tactical Operation Center (CTOC) and the Army Forces Command Post (ARFCP). (The ARFCP, CTOC, DASC, and TASE essentially form the Corps Headquarters.)



JTF : Joint Tosk Force
JOC : Joint Operotions Center
AFFCP : Air Force Forces Command Post
ARFCP : Army Forces Commond Post
TACC : Tacticol Air Control Center
CTOC : Corps Tactical Operotions Center
DASC : Direct Air Support Center
TASE : Tactical Air Support Element
MIBAR : Military Intelligence Battolion (Airborne Reconnoissance)
FSCC : Fire Support Control Center

FACP : Forward Air Control Post
ASRT : Air Support Radar Team
RP : Reporting Post
CRP : Control & Reporting Post
CRC : Control & Reporting Center
GLO : Ground Liaison Officer
ARLO : Airborne Reconnaissance Liaison Officer
$TACP_1$: ALO + FAC + Communication Personnel
$TACP_2$: ALO + Communicotion Personnel

Fig. 10—Present Tactical Air Control System

The Air Force would also supply Tactical Air Control Parties (TACPS) at battalion, brigade, and division level. In Fig. 10, the TACPS that consist of an Air Liaison Officer (ALO) and Forward Air Controller (FAC) plus requisite communication personnel are designated by the subscript "1." The subscript "2" indicates a TACP of only an ALO and necessary communication personnel. A $TACP_1$ is attached to each battalion, while a $TACP_2$ is attached to each brigade and division headquarters. In some instances the FAC, and possibly his radio operator, may operate from a light reconnaissance aircraft.

Figure 10 also shows a joint task force (JTF) command element and the associated joint operation center (JOC), (lower left corner) as well as tactical fighter elements, and tactical reconnaissance elements (lower right corner). Tactical airlift elements are not shown, but their use when required is implied.

CURRENT COMMUNICATIONS

Figure 11 outlines some of the communication links of today's Tactical Air Control System. Figure 12 shows some of the communication links for the present Army system. The command and control elements show in Figs. 11 and 12 correspond to those in Fig. 10. (The communication links were omitted from Fig. 10 for clarity of presentation.)

In today's system, preplanned requests for missions are apt to come from the air liaison officers (ALOs) at battalion, brigade, and division command posts. These requests reflect cooperative and coordinated planning between Army commanders and their Air Force advisors (ALOs) at these various levels of field command. Usually a period of about 8 to 24 hours, or more, may elapse between the time of a preplanned request and execution of the mission. Immediate requests for air support can be made by the Forward Air Controllers (FACs) and ALOs at battalion level and the ALOs at brigade and division headquarters.

For a "conventional" division there will be about nine FACs and thirteen ALOs. Thus, it is possible that requests for immediate air support could come from about twenty-two different message sources in a "conventional" division. Carried one step further, this implies that for a corps consisting of three divisions on the line, as many as

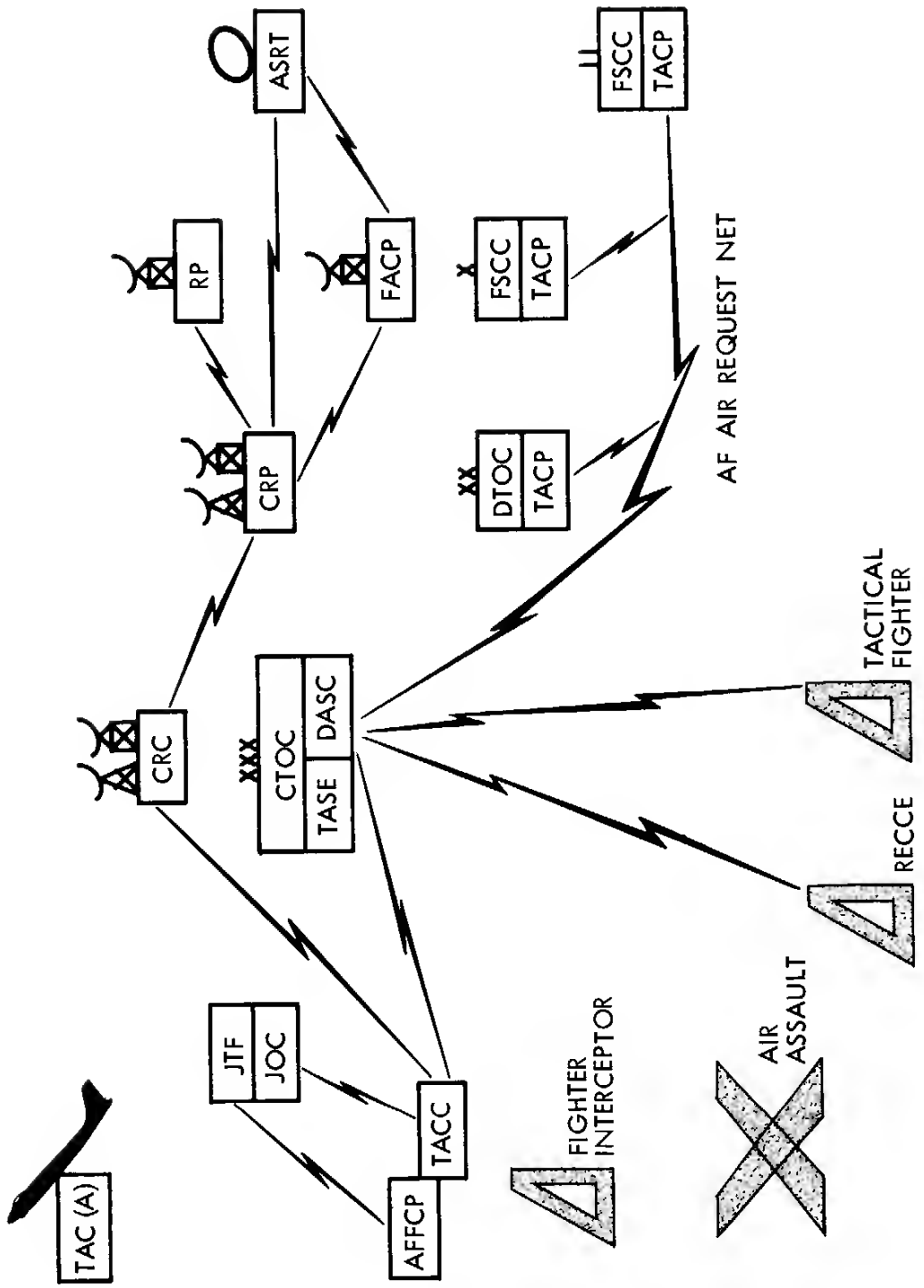


Fig. 11—Tactical air control system (TACS)

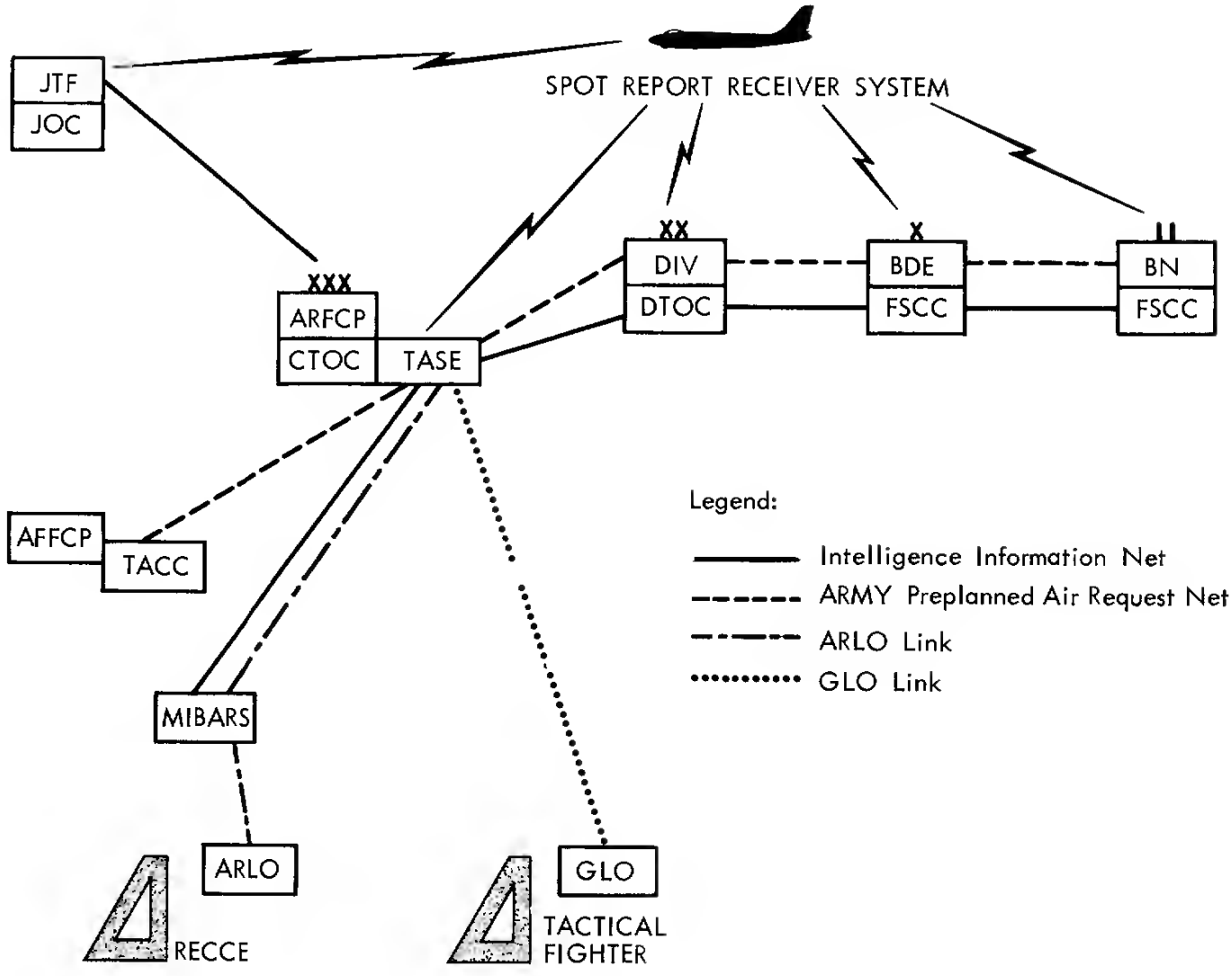


Fig. 12—Army air ground system (AAGS)

sixty-six different sources could conceivably be generating requests for close air support, reconnaissance, air defense and airlift.

SIMULATION OF AN AUTOMATED TACTICAL AIR CONTROL SYSTEM

The potential assistance that data automation could provide to the Tactical Air Control System just described is evident. Data automation could handle much of the message routing, status updating, etc. In short, many of the tasks that are now performed manually could doubtless be done faster and more accurately, and many tasks considered infeasible today should become possible.

A logical first step in acquiring field service data automation for the Tactical Air Control System, is the development of a digital message entry device suitable for use by Forward Air Controllers. The U.S. Air Force tested a Digital Message Entry System (DMES) in early 1966 with generally favorable results.[5]

It is to be expected that the next step in the development process will be the employment of a computer to process the requests for resources in terms of the resources available. In the case of FAC requests for close air support (CAS), the decision point is the Fighter Duty Officer (FDO) in the DASC. It is this facet of an Automated Tactical Air Control System--FAC, BASE, and DASC--that has been simulated using the JOSS System and will be discussed below.

The digital message entry device tested at the Tactical Air Warfare Center in 1966 is shown in Fig. 13. The message format of this device has been used in the JOSS simulation. Provisions have been made in the simulation programs for requests for resources from six FACs and status of resources from four airbases, as shown in Fig. 14. A single console serves the DASC FDO. The DASC program monitors the store-and-forward communication system, looking for requests from the FACs. When a request is received, the FAC message is automatically printed, followed by a listing of the resources available at the four airbases. The FDO is then afforded an opportunity to assign flights of aircraft to meet the request. Flights may be assigned from any or all airbases. Once the FDO has indicated an end to the assignment of resources, the program automatically sends the assignments and the FAC request (for

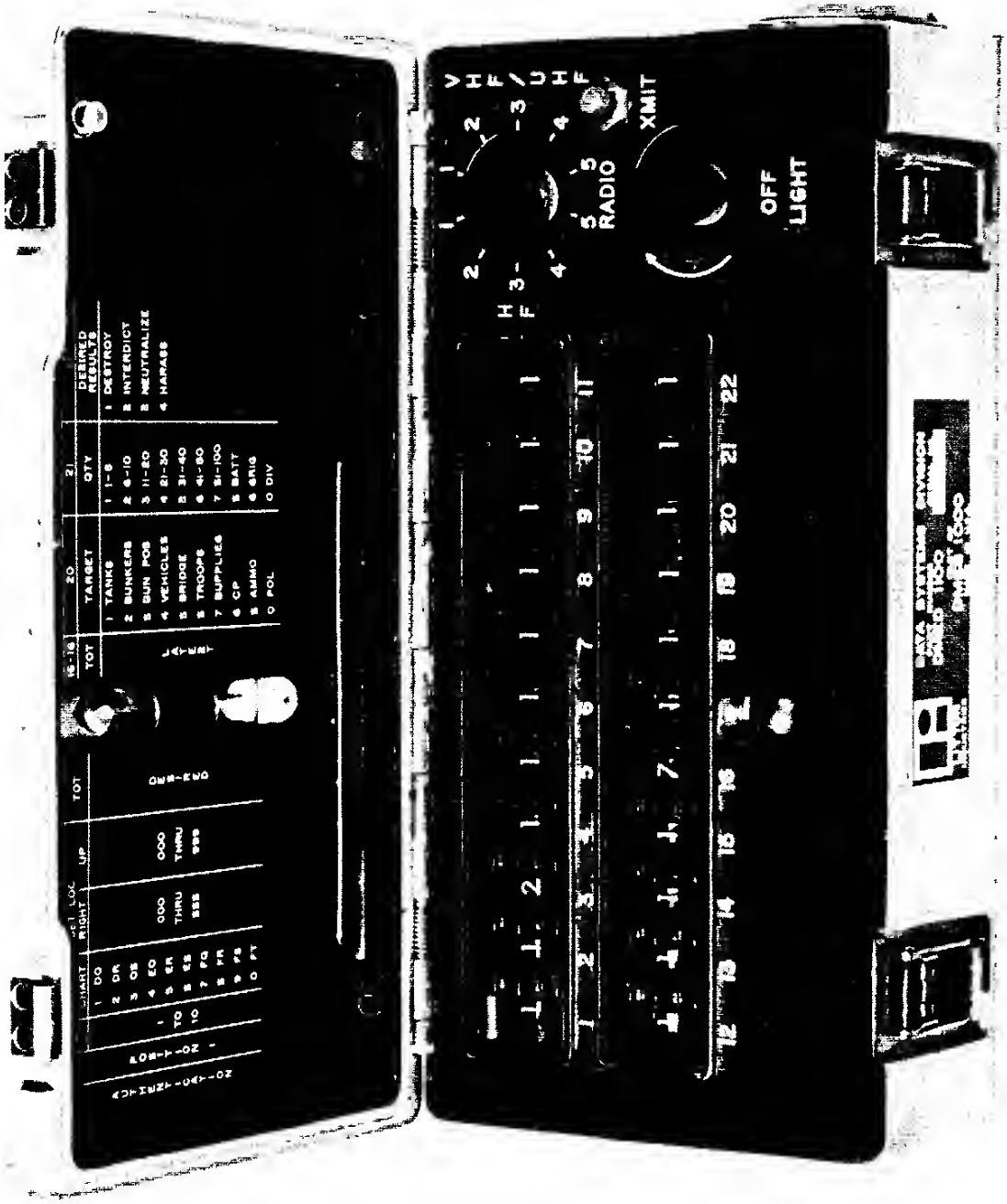
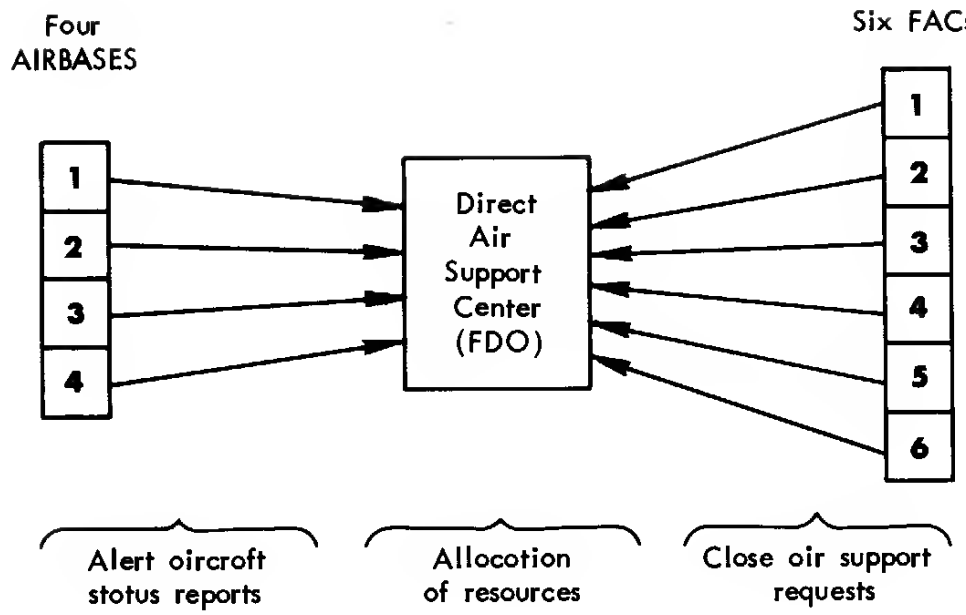


Fig. 13—Litton digital message entry device (DMED)



Features:

- Eleven JOSS consoles used
- Keyboard input
- Hard copy output

Fig. 14—JOSS ATACS simulation

information purposes) to all appropriate airbases. An acknowledgment, containing all flights assigned, is also sent to the requesting FAC. These features are illustrated in Fig. 15.

At the airbases where an assignment has been made, the status of available aircraft is automatically updated, so that the next time the DASC receives a request, the latest status of available resources will no longer have the previously assigned flights showing.

Once the FAC has received his acknowledgment from the DASC, his program automatically prepares for him a blank CAS request form which can be filled in with pencil before formally inserting the message into the system.

As initially programmed, the ATACS simulation uses all 25 file-items in a single JOSS file, allocated as shown in Fig. 16. The store-and-forward communications portion of the simulation--in this case handling only numerical data--occupies 20 of the file items, with the DASC, BASE, and FAC programs and other housekeeping features filling the remaining 5 file-items. The various interconnections of the 11 JOSS consoles through the medium of the common file are shown in Fig. 17.

AN ILLUSTRATION OF THE ATACS SIMULATION

The system simulation just outlined is easily illustrated by running through a sequence of operations that might occur at the various consoles. Figure 18 shows the close air support form filled in by the FAC, both in pencil and then input to the console. The FAC simulation program is designed to offer the FAC as many opportunities as needed to make corrections in the message, since human error at this point could result in a misallocation of resources.

The BASE program has routines that permit operation in a DASC monitoring mode, or a status input mode, or a status correction mode. Figure 19 shows the "educational" routine that can be called out by a new operator to provide background information on the operation of the programs. Normally, of course, all this information is not necessary for a well-practiced operator. Figure 20 presents an airbase status report consisting of three flights that has been inserted, checked, and stored in the status file-item for Nha Trang airbase. As in the FAC

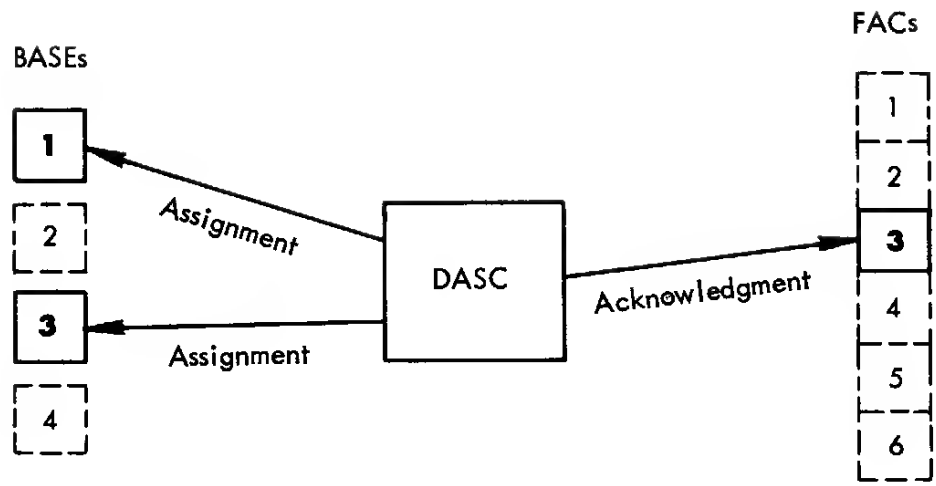


Fig. 15—ATACS automatic AIRBASE assignments and FAC acknowledgment

FILE 232 (SYOP2)

FAC "TRIGGERS"	1	k = 0,1	
	2	k = 0,2	
	3	k = 0,3	
	4	k = 0,4	
	5	k = 0,5	
	6	k = 0,6	
AIRBASE STATUS	7	"B-LIST"	(TAN SON NHUT)
	8	"	(NHA TRANG)
	9	"	(BIEN HOA)
	10	"	(CAM RANH BAY)
FAC REQUESTS	11	"A-LIST"	
	12	"	
	13	"	
	14	"	
	15	"	
	16	"	
AIRBASE ASSIGNMENTS	17	"C-LIST"	(TAN SON NHUT)
	18	"	(NHA TRANG)
	19	"	(BIEN HOA)
	20	"	(CAM RANH BAY)
PROGRAMS, ETC.	21	TEMP	
	22	DASC	
	23	BASE	
	24	FAC	
	25	DATA	

Fig. 16—JOSS ATACS file usage

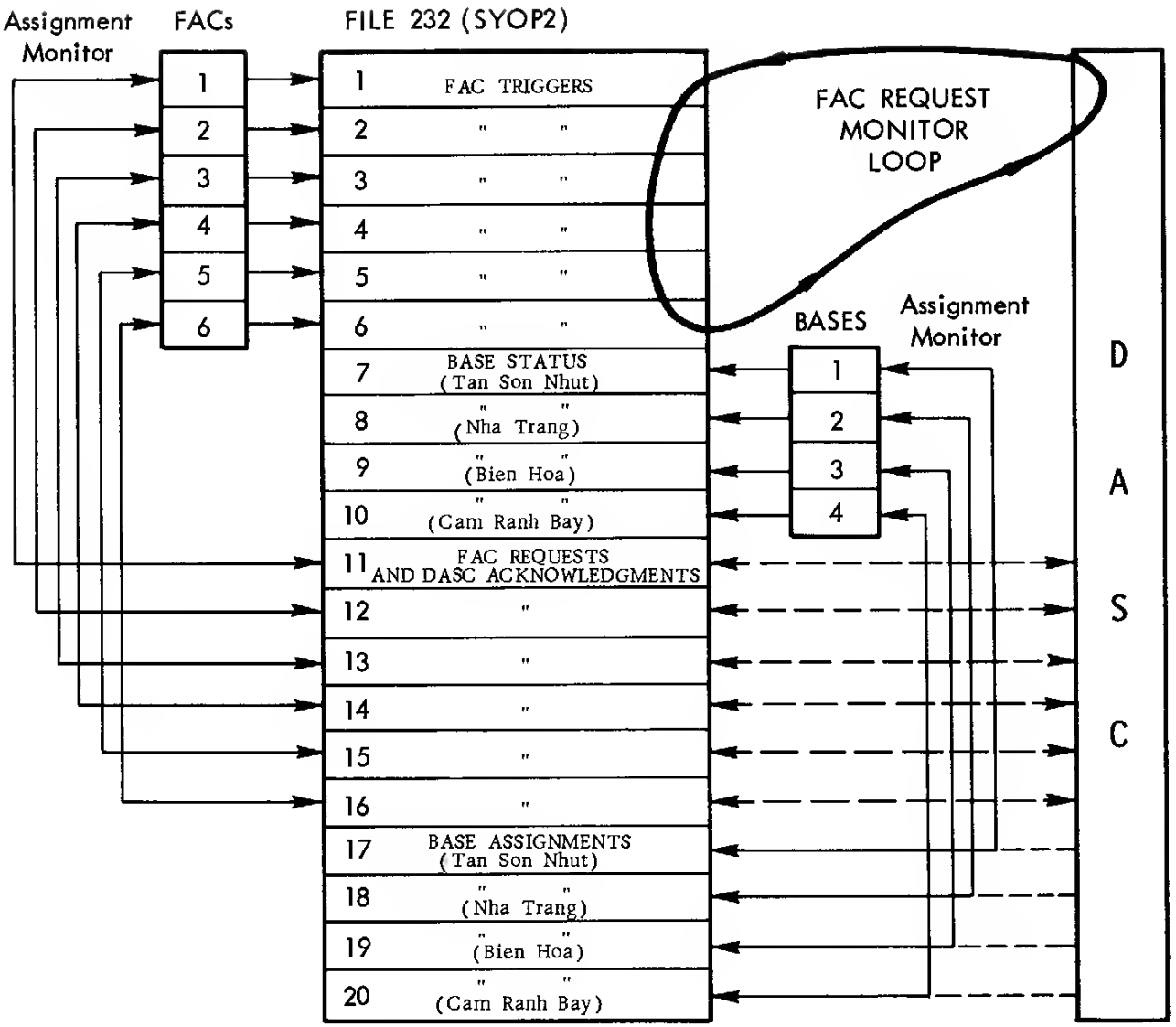


Fig. 17—JOSS ATACS message switching

(1)

00247 AIR SUPPORT REQUEST NO. 4

Target Location No. 5 [1=XU 2=YU 3=ZU 4=AP 5=BP 6=XT 7=YT 8=ZT 9=AN 0=BN]
Target Coordinates: 473 RIGHT 281 UP

TIME ON TARGET [Desired] 0900 TIME ON TARCET [Latest] 0945

TARGET TYPE 4 QUANTITY 3
1=Tnks 2=Bnkrs 3=Gun Pos 4=Vehcls 5=Brdge 6=Trps 7=Supls 8=CP 9=Ammo 0=POL
1=1-5 2=6-10 3=11-20 4=21-30 5=31-40 6=41-50 7=51-100 8=BATT 9=BRIC 0=DIV
DESIRED RESULTS 1 [1=DESTROY 2=INTERDICT 3=NEUTRALIZE 4=HARASS]

WHEN READY TO INPUT NEW REQUEST, TYPE 'K = 1'.
K = 1
Code; M. Type; M. No.
A(1) = 47
A(2) = 1
A(3) = 4
UTM; Rt.; Up:
A(4) = 5
A(5) = 473
A(6) = 281
Time:
A(7) = 0900
A(8) = 0945
Tgt.: Type; Quan.; Res.
A(9) = 4
A(10) = 3
A(11) = 1

For corrections, type 'K = 1', otherwise, 'K = 0'.
K = 0

00247 AIR SUPPORT REQUEST NO. 4

TGT. LOC. No.= 5 COORDS: 473 RIGHT 281 UP

TIME ON TARCET [Desired] 900 TIME ON TARGET [Latest] 945

TGT. TYPE-4 QUANTITY-3 DESIRED RESULTS-1

For corrections, type 'K = 1', otherwise, 'K = 0'.
K = 0

Type your FAC Number to input FAC REQUEST.

k = 1
Trigger No. 1 is set. File-item 11 is updated.

Fig. 18—Forward Air Controller request for close air support

```
Do part 17
         AIRBASE IDENTIFICATION (START-UP) PROGRAM
For AIRBASE IDENT, type 'k=1', otherwise, 'k=0'.
           k = 1
AIRBASE IDENTIFICATION
Tan Son Nhut Airbase:  Set f = 1.
Nha Trang Airbase:     Set f = 2.
Bien Hoa Airbase:      Set f = 3.
Cam Ranh Bay Airbase:  Set f = 4.
           f = 2

For STATUS REPORT key, type 'k = 1', otherwise, 'k = 0'.
           k = 1
KEY TO AIRCRAFT STATUS REPORT
l = Beginning line number.(Don't confuse 'l' with '1'.)
L = Last line number. (L-l+1 = the "line block size.")
T = Time of STATUS REPORT
f = Airbase Index
n = Line Index

B(f,n,1) = Data Line Number
B(f,n,2) = Squadron Number
B(f,n,3) = Mission Identification Number
B(f,n,4) = Number of Aircraft in Alert Flight
B(f,n,5) = Type Aircraft in Alert Flight
B(f,n,6) = Call Sign Identification Number
B(f,n,7) = Time Aircraft Go On Alert Status
B(f,n,8) = Mission Type Identification Number
B(f,n,9) = Ordnance Load Identification Number

Type 'k = 1' for key to CALL SIGN,MSN TYPES,ORD LOAD.
           k = 1
KEY TO CALL SICNS, MSN TYPES, ORDNANCE LOADS
Call Signs: 1 = RED; 2 = WHITE; 3 = BLUE; 4 = CREEN; 5 = GOLD
Mission Types: 1 = INTERDICT; 2 = CAS; 3 = AIR DEF; 4 = RECCE
      ORDNANCE LOADS
       1 = Cuns, AIM-9/B
       2 = Guns, Napalm
       3 = Guns, 750 lb CP Bombs
       4 = Guns, 2.75 in Rockets
       5 = Cuns, Napalm, 2.75 in Rockets
       6 = Cuns, Napalm, 750 lb GP Bombs
       7 = Guns, 750 lb CP Bombs, 2.75 in Rockets
       8 = Guns, CBU
       9 = Guns, Napalm, CBU
      10 = Cuns, ACM-12B
      11 = Cuns, 500 lb Frag Bombs
      12 = Guns, Napalm, 500 lb Frag Bombs


To input a STATUS REPORT, type 'k=1',otherwise, 'k=0'.
           k = 1
```

Fig. 19—BASE start-up and "educational" printout

```
AIRBASE STATUS PROGRAM (New Data Blocks or Corrected Lines)
What is beginning line number?
          l = 1
What is last line number?
          L = 3

Line No.:
    B(f,n,1) =        1
Sqdn No.:
    B(2,1,2) = 436
Msn ID No.:
    B(2,1,3) = 1
No. A/C:
    B(2,1,4) = 3
Type A/C:
    B(2,1,5) = 4
Call Sign:
    B(2,1,6) = 3
Alert Time:
    B(2,1,7) = 0800
Msn Type:
    B(2,1,8) = 2
Ord Load:
    B(2,1,9) = 12

  1   436    1    3 F-  4    3    800    2    12
Corrections? Type 'k = 1', else 'k = 0'.
          k = 0

Line No.:
    B(f,n,1) =        2
Sqdn No.:
    B(2,2,2) = 436
Msn ID No.:
    B(2,2,3) = 2
No. A/C:
    B(2,2,4) = 2
Type A/C:
    B(2,2,5) = 4
Call Sign:
    B(2,2,6) = 5
Alert Time:
    B(2,2,7) = 0900
Msn Type:
    B(2,2,8) = 2
Ord Load:
    B(2,2,9) = 9

  2   436    2    2 F-  4    5    900    2     9
Corrections? Type 'k = 1', else 'k = 0'.
          k = 0

Line No.:
    B(f,n,1) =        3
```

Fig.20—BASE alert flight status input

```
Sqdn No.:
    B(2,3,2) = 436
Msn ID No.:
    B(2,3,3) = 3
No. A/C:
    B(2,3,4) = 4
Type A/C:
    B(2,3,5) = 4
Call Sign:
    B(2,3,6) = 6
Alert Time:
    B(2,3,7) = 1000
Msn Type:
    B(2,3,8) = 2
Ord Load:
    B(2,3,9) = 7

  3   436    3    4 F-  4    6   1000    2     7
Corrections? Type 'k = 1', else 'k = 0'.
           k = 0

Insert TIME of this STATUS REPT.
           T = 1432

Set lower limit = l and upper limit = L
on data to be transmitted and/or displayed.
           l = 1
           L = 3

Is complete formatted copy desired before transmitting data?
If so, type 'k = 1', otherwise, 'k = 0'.
           k = 1

ALERT AIRCRAFT STATUS REPT:  NHA TRANG Airbase

           STATUS REPORT TIME   1432  HRS

(1)   (2)  (3)  (4)  (5)   (6)   (7)   (8)   (9)
LINE  SQD  MSN  NO.  TYPE  CALL  TIME  MSN   ORDN
NO.   NO.  NO.  A/C   A/C  SIGN  AVAIL TYPE  LOAD

  1   436    1    3 F-  4    3    800    2    12
  2   436    2    2 F-  4    5    900    2     9
  3   436    3    4 F-  4    6   1000    2     7

Corrections? Type 'k = 1', else 'k = 0'.
           k = 0
THIS DATA HAS BEEN STORED IN FILE-ITEM  8 (BASE).



THIS MACHINE IS NOW IN THE DASC MONITORING LOOP.
```

Fig. 20 (cont.)—BASE alert flight status input

program, opportunities to correct the flight status data input occur repeatedly. This input program can also be used to add lines of data (flights of aircraft) to an already existing status report; hence, the reason for asking for values of the parameters "1" and "L" both at the beginning and end of the data input.

Keeping the illustrated request for resources and status of resources at one airbase in mind, it is now possible to shift to the DASC program and view the result at the decision-making point. Figure 21 shows the printout of the FAC request of Fig. 18. The figure also illustrates that the aircraft resources at Nha Trang airbase are in accord with the input shown in Fig. 20. The Nha Trang status demonstrates one of the decision aids afforded the FDO: namely, a flight of aircraft that is not available in time to satisfy the request is listed as "Not Available till 1000 hrs," where the time depends on the input from the airbase.

The FDO is given the choice between making an assignment or going to the next FAC request. If he chooses to make the assignment, the aircraft assignment subroutine is entered and the FDO assigns aircraft flights from any or all airbases until he deems that sufficient resources have been allocated.* The assignment subroutine sends assignment messages in turn to each chosen base. At the end, when the FDO has finished, an acknowledgment message is sent to the FAC, giving him the data on all flights that were assigned to fulfill his request. The DASC program then returns automatically to the interrogation of FAC triggers to see if other messages are in the queue.

The assignment message sent to Nha Trang airbase is shown in Fig. 22. The assigned flights and the FAC message are received at the airbase console. It is apparent that, at this point, the only human input is an update of actual time (which will appear at the DASC the next time status is requested).** The lines of data for the assigned flights are deleted from the status report, and the data lines are renumbered using

---

*There is an obvious opportunity here to use a computerized model for "best" resource allocation. That feature has not yet been added to the simulation.

**Eventually, the JOSS System software will be modified to make this operation possible under program control. Then this assignment and updating part of the BASE program will be totally automatic.

```
MESSAGE FROM FAC No.  1

For FULL FORMAT, type 'k=1', for SHORT FORMAT, 'k=0'.
          k = 1

00247  AIR SUPPORT REQUEST NO.   4

Target Location No.  5 [1=XU 2=YU 3=ZU 4=AP 5=BP 6=XT 7=YT 8=ZT 9=AN 0=BN]

Target Coordinates:  473 RIGHT    281 UP

TIME ON TARGET [Desired]  900     TIME ON TARGET [Latest]  945

TARGET  TYPE   4       QUANTITY  3
1=Tnks 2=Bnkrs 3=Gun Pos 4=Vehcls 5=Brdge 6=Trps 7=Supls 8=CP 9=Ammo 0=POL
1=1-5 2=6-10 3=11-20 4=21-30 5=31-40 6=41-50 7=51-100 8=BATT 9=BRIG 0=DIV

DESIRED RESULTS  1   [1=DESTROY  2=INTERDICT  3=NEUTRALIZE  4=HARASS]


           AVAILABLE AIRCRAFT

LINE  SQD  MSN  NO.  TYPE  CALL  TIME  MSN   ORDN
NO.   NO.  NO.  A/C   A/C  SIGN  AVAIL TYPE  LOAD

TAN SON NHUT     TIME: 1230 hrs.
  1   224    1    4 F-100    1    500    1     1
  2   501    3    1 F-100    2    800    2     2
  3   502    4    2 F-100    3    810    2     2
  4   503    5    3 F-100    2    820    2     2
  5   504    6    4 F-100    2    830    2     2

NHA TRANG        TIME: 1432 hrs.
  1   436    1    3 F-  4    3    800    2    12
  2   436    2    2 F-  4    5    900    2     9
  3  N. A. till 1000 hr

BIEN HOA         TIME: 1259 hrs.
  1   502    7    3 F-100    1    455    2     1
  2   412   11    4 F-100    2    455    2     2
  3   327   12    2 F-100    3    455    2     3

CAM RANH BAY     TIME:  750 hrs.
  1   207    4    2 F-100    4    545    2     3


To assign aircraft, type 'k = 1', otherwise 'k = 0'.
          k = 1

           AIRCRAFT ASSIGNMENT PROGRAM
```

Fig.21—DASC response to FAC request

```
Select AIRBASE  [f=1 TSN   f=2 NT   f=3 BH   f=4 CRB].
           f = 1
Select the NUMBER OF FLIGHTS:
    C(1,0,0) = 1
TO ASSIGN FLIGHTS, INSERT 'DATA LINE NUMBERS.'
    C(1,0,1) = 3
TAN SON NHUT AIRBASE ASSIGNMENT TRANSMITTED


     For more AIRBASES, type desired "f", otherwise, 'f=0'.
           f = 2
Select the NUMBER OF FLIGHTS:
    C(2,0,0) = 1
TO ASSIGN FLIGHTS, INSERT 'DATA LINE NUMBERS.'
    C(2,0,1) = 2
NHA TRANG AIRBASE ASSIGNMENT TRANSMITTED


     For more AIRBASES, type desired "f", otherwise, 'f=0'.
           f = 0


FAC-1 TRIGGER = 0
FAC-1 REQUEST SATSIFIED
```

Fig.21 (cont.)—DASC response to FAC request

```
      DASC MESSAGE FOLLOWS:

THE FOLLOWING FLIGHTS ARE ASSIGNED CLOSE AIR SUPPORT

LINE  SQD  MSN  NO.  TYPE  CALL  TIME  MSN   ORDN
NO.   NO.  NO.  A/C   A/C  SIGN  AVAIL TYPE  LOAD
  2   436    2    2 F-  4     5    900    2     9


Now Updating Status Report.  Give Present TIME:
           T = 1446
This BASE has  2 flights remaining.
UPDATED STATUS REPORT HAS BEEN STORED IN FILE-ITEM  8 AT T = 1446 HRS



FAC CLOSE AIR SUPPORT REQUEST FOLLOWS:

00247  AIR SUPPORT REQUEST NO.   4

Target Location No.  5 [1=XU 2=YU 3=ZU 4=AP 5=BP 6=XT 7=YT 8=ZT 9=AN 0=BN]

Target Coordinates:  473 RIGHT    281 UP

TIME ON TARGET [Desired]  900     TIME ON TARGET [Latest]  945

TARGET  TYPE   4       QUANTITY  3
1=Tnks 2=Bnkrs 3=Gun Pos 4=Vehcls 5=Brdge 6=Trps 7=Supls 8=CP 9=Ammo 0=POL
1=1-5 2=6-10 3=11-20 4=21-30 5=31-40 6=41-50 7=51-100 8=BATT 9=BRIG 0=DIV

DESIRED RESULTS  1   [1=DESTROY  2=INTERDICT  3=NEUTRALIZE  4=HARASS]

 END DASC Message

Now in DASC MONITORING LOOP.
I'm at step 20.006.
```

Fig. 22—DASC assignment message received by BASE

a file "push-up" subroutines. The BASE console then reverts to monitoring the DASC assignment file-items awaiting the next assignment of resources.

Figure 23 shows the acknowledgment sent to the FAC. After the acknowledgment has been received, the FAC program prepares a new close air support request format, as shown previously in Fig. 18.*

For testing or demonstrating a system simulation such as this, it is desirable to have "canned" data available to be input in place of inputs that would normally derive from operators at consoles. A "start-up" subroutine is available as part of the DASC program. It informs the user of its progress, as indicated in Fig. 24. Once all data is inserted, the FAC trigger interrogation is begun and the six FAC requests can be executed.

EXERCISING THE SIMULATED ATACS

To work out minor program bugs and ensure program compatibility, the JOSS ATACS simulation was exercised in about 15 two-hour sessions by five UCLA Army ROTC students. While it was not the express purpose of these exercises to make observations about desirable characteristics of an automated tactical air control system simulation, some conclusions became very apparent as the exercises progressed. For example,

- o No difficulty was encountered in training the ROTC students in operating the JOSS hardware, learning the operational procedures of using the programs, and shifting among the DASC, BASE, and FAC positions.
- o It was quickly discovered that the Digital Message Entry Device format was too inflexible to permit inclusion of all the information that a FAC might want to include in a single message, e.g., the format does not permit stating

---

*The observant reader may have noted that all the illustrations of this simulation were carried out in a time-sequential fashion, on a single JOSS console. This capability was intentionally designed into the programs. In a normal simulation run, all consoles (up to a total of eleven) would be operating in parallel. However, the sequential operation feature is very useful, since a single operator can switch from one program to another, either to demonstrate the entire system using a single console, or to make and check out program changes.

```
Do part 1000
Entering DASC Monitoring Loop. What FAC No. are you?
          k = 1


THIS MACHINE NOW AWAITINC DASC RESPONSE

(To input NEW REQUEST, "Interrupt" and type "Do part 10.")


DASC REPLY TO FAC-1, MESSAGE NO.  4
FOLLOWING FLIGHTS ASSIGNED:

LINE  SQD  MSN  NO.  TYPE  CALL  TIME  MSN   ORDN
NO.   NO.  NO.  A/C   A/C  SIGN  AVAIL TYPE  LOAD

TAN SON NHUT:
  3   502    4    2 F-100    3    810    2     2

NHA TRANG:
  2   436    2    2 F-  4    5    900    2     9

COOD LUCK
```

Fig. 23—DASC acknowledgement message received by FAC

```
Use file 232 (syop2).
Roger.

Recall item 22 (DASC).
Done.

Do part 100.
DASC PROGRAM START-UP:

FAC No. 1 Trigger Set
FAC No. 2 Trigger Set
FAC No. 3 Trigger Set
FAC No. 4 Trigger Set
FAC No. 5 Trigger Set
FAC No. 6 Trigger Set


This Program "fetches"Data from File 234 (syop4)
and places it in BASE Items 7,8,9,10 - File 232 (syop2).
BASE No. 1 Status Updated.
BASE No. 2 Status Updated.
BASE No. 3 Status Updated.
BASE No. 4 Status Updated.
Data Fetch Program Finished.

FAC No. 1 Message In
FAC No. 2 Message In
FAC No. 3 Message In
FAC No. 4 Message In
FAC No. 5 Message In
FAC No. 6 Message In
All BASE Data Updated. All FAC Messages Inserted.


DASC PROGRAM BEGUN.  This machine will continue
in a LOOP until FAC message is received, or until
stopped by "INTERRUPT" action.
```

Fig.24—ATACS simulation "canned" demonstration start-up

in a single request the number of troops, gun positions, bunkers, vehicles, etc., that might all be associated with a single target area.*

- The exercises fully illustrated the axiom that "the display of information can be one of the principal inherent pitfalls in any on-line computer-aided decision system."

  It was found that the time required to receive and display a FAC request, display the BASE status reports, and allocate resources to satisfy the request required from 2-1/2 to 5 minutes. Much of this time was occupied by program-controlled typing. Even with the JOSS console Selectric typewriter, character-at-a-time display of information is exceedingly time consuming, in comparison with other actions required in the decision process.

  One concludes that it would be desirable to have information displayed simultaneously to the FDO on three separate CRT displays: one for the FAC request message, one for the BASE status information, and the third for the assignment message being composed by the FDO.

  A second conclusion is that an "overlay" keyboard--thus converting each console from general to special purpose--would be highly desirable.

- The JOSS simulation of the close air support feature of an ATACS was deemed sufficiently adequate to make further system simulation of considerable value in developing experience and software prior to establishing requirements for the CAS feature of a field service computer-based system. JOSS-like on-line, time-shared systems offer the advantage that the various functional elements of an ATACS--close air support, reconnaissance, interdiction, air defense, and air lift/air assault--can each be programmed and tested using the same computer equipment. Of course, it is likely that the JOSS System per se might have trouble supporting all of these functions simultaneously in a complete, real-time simulation.

---

*It has long been recognized that even an automated tactical air control system should have provision for voice communication links and voice data inputs also. (See Refs. 3, 4.) While it is not surprising that this simulation exercise reinforces that view, it was disturbing to realize how much effort would have to be spent at the DASC to hand-insert in the DASC console the voice-transmitted additional information that would be needed to make use of computer assistance in the form of resource allocation algorithms, etc.

VI. SUMMARY

This paper has described several techniques and applications of RAND's on-line, time-shared JOSS multiconsole computer system to various facets of gaming and simulation. The JOSS System software characteristics that make possible the store-and-forward communications feature among consoles--namely, common sharing of files and indirect file operations--have only been available for seven months. During that time independent efforts at RAND have established the feasibility of using the JOSS System in multiconsole gaming and simulations by programming a strategic diplomatic exchange game and exercising a tactical battle game of the text-only and data-only type, respectively, and by programming and exercising a system simulation involving the classic triad of resource supply, requests for resources, and resource allocation, specifically applied to the close air support function of a hypothetical automated tactical air control system.

Thus far, the principal result of this effort has been to gain experience and prepare and check out programs basic to future gaming and simulation activities of considerably larger scope. Even cursory experience with the hypothetical ATACS simulation has been sufficient to suggest that certain inadequacies in the DMED message format have been encountered and that there is considerable effectiveness at low cost in using a JOSS-like system to gain the insight and experience needed to properly specify requirements for computer hardware and software for systems such as an ATACS.

The ease in programming afforded by a JOSS-like language ensures that system users can make direct contributions to the game or simulation programming effort, both during the initial phases of writing programs and in the later--and often crucial--phases of revising and improving programs. The involvement of at least some of the game or simulation users at the basic programming level may result not only in greater understanding of the game/simulation structure by the users, but also may contribute considerably to reducing the amount of time needed to generate programs and program modifications. These features could result in a step-function improvement in game/simulation efforts

in the future, for, in general, these are features that have not been present in the past.

It is not the intent of this paper to leave the reader with the impression that an on-line, time-shared JOSS-like computer system is a panacea for all the difficulties associated with computer-aided gaming and simulation. RAND's JOSS consoles are definitely output-limited in terms of the rate of presentation of information versus the ability of the human to absorb and use such information. But the mismatch is not great. If very extensive programs for numerical data processing and/or table look-up are required in running the game/simulation, experience indicates that, naturally, time-shared computers cannot match straight batch processing systems of comparable characteristics.

But the time-shared JOSS-like computer offers the opportunity to interconnect participants in a game/simulation without regard to location. For example, Type 33 and 35 Teletype consoles can, with certain minor limitations, be used as JOSS consoles. Conventional teletype or data-phone connections permit the internetting of JOSS consoles throughout the country. Features such as this, coupled with built-in store-and-forward communications potential, a simple and convenient I/O capability, and an easily mastered programming language may make on-line, time-shared multiconsole computers an integral and important part of many future games and simulations.

Appendix A

JOSS PROGRAMS FOR A TWO-PARTY, TEXT-ONLY WAR GAME

```
Type part 10.
10.01 Type "JOSS Store-and-Forward COM-SYSTEM Start-up Program.".
10.02 Type _,_.
10.03 Type "Give item no. to which your messages are to be transmitted.".
10.031 Demand T.
10.04 Line.
10.05 Type "Give item no. from which you will receive messages.".
10.051 Demand R.
10.06 Line.
10.07 Type "To begin operation in the RECEIVE mode, type K = '0'.".
10.071 Type "To begin operation in the TRANSMIT mode, type K = '1'.".
10.0711 Demand K.
10.0712 Type "ERROR. K must be '0' or '1'." if K>1 or K<0.
10.0713 To step 10.07 if K>1 or K<0.
10.072 Line.
10.08 Type "END OF START-UP PROGRAM.".
10.081 Do part 2 if K = 0.
10.082 Do part 100 if K = 1.
```

Fig. 25— Start-up program for two-party, text-only war game

```
Type part 2.
2.0001 Type "Entering the MESSAGE MONITORING mode.".
2.001 Recall item R.
2.002 To step 2.2 if k ≠ 0.
2.003 Do step 2.005 for n = 1(1)5000.
2.0031 Type "No message.".
2.004 To step 2.001.
2.005 Set k = k.
2.2 Page.
2.22 Type "MESSAGE RECEIVED:".
2.23 Type _,_.
2.3 Type part 1 if k ≠ 0.
2.4 Discard item R.
2.41 Delete part 1.
2.5 Set k = 0.
2.6 File k as item R.
2.9 Type "END OF MESSAGE.".
2.91 Type _,_,_.
2.911 Type "To REPLY to above message, set K = '1',".
2.912 Type "To continue in MESSAGE Monitoring mode, set K = '0'.".
2.913 Demand K.
2.914 Do part  100 if K = 1.
2.915 To step 2.92 if K = 0.
2.92 Type "Back in message monitoring loop.".
2.99 To step 2.001.
```

Fig. 26—Message monitoring program for two-party, text-only war game

```
Type part 100.
100.001 Page.
100.002 Type "MESSAGE GENERATION AND TRANSMISSION PROGRAM.".
100.003 Type _,_,_.
100.1 Set k = 1.
100.2 Type "Preface each line of message with a "Part 1" number.".
100.3 Type "After composing message, type "Go.".".
100.4 Stop.
100.41 Type part 1.
100.42 Type _,_.
100.43 Type "To make CORRECTIONS, set C = '1', else set C = '0'.".
100.44 Demand C.
100.45 To step 100.5 if C = 0.
100.46 Type "Make CORRECTIONS, then type 'Go.'.".
100.47 Stop if C = 1.
100.48 Type "To TRANSMIT message, set K = '1'.".
100.481 Type "To CHECK message again for corrections, set K = '0'.".
100.482 Demand K.
100.483 To step 100.41 if K = 0.
100.484 Line if K = 1.
100.5 Discard item T.
100.6 File k, part 1 as item T.
100.61 Type "MESSAGE TRANSMITTED.".
100.7 Delete part 1.
100.8 Type "To shift to MESSAGE MONITORING mode, set K = '1'.".
100.8001 Type "To send ANOTHER MESSAGE, set K = '0'.".
100.801 Demand K.
100.81 Do part 2 if K = 1.
100.82 To step 100.001 if K = 0.
```

Fig. 27—Message generation and transmission program for two-party, text-only game

Appendix B

JOSS PROGRAMS FOR A BLUE, RED, AND GREEN TEXT-ONLY WAR GAME

```
Type part 10.
10.01 Type "JOSS Store-and-Forward COM-SYSTEM Start-up Program.".
10.02 Type _,_.
10.03 Type "Give item no. for storing messages to your opponent.".
10.031 Demand T(1).
10.032 Type "Give item no. for storing message copies for GREEN.".
10.033 Demand T(2).
10.04 Line.
10.05 Type "Give item no. for receiving messages from your opponent.".
10.051 Demand R(1).
10.052 Type "Give item no. for receiving messages from GREEN.".
10.053 Demand R(2).
10.06 Line.
10.07 Type "To begin operation in the RECEIVE mode, type K = '0'.".
10.071 Type "To begin operation in the TRANSMIT mode, type K = '1'.".
10.0711 Demand K.
10.0712 Type "ERROR. K must be '0' or '1'." if K>1 or K<0.
10.0713 To step 10.07 if K>1 or K<0.
10.072 Line.
10.08 Type "END OF START-UP PROGRAM.".
10.081 Do part 2 if K = 0.
10.082 Do part 100 if K = 1.
```

Fig. 28—BLUE/RED start-up program

```
Type part 2.
2.0001 Type "Entering the MESSAGE MONITORING mode.".
2.0002 Type "Checking for message from GREEN.".
2.00021 Set X = 0.
2.0003 Recall item R(2).
2.00031 Type "No message from GREEN." if k = 0.
2.00032 Discard item R(2) if k = 1.
2.0004 Set X = k.
2.0005 To step 2.001 if k = 0.
2.0006 Page.
2.0007 Type "MESSAGE RECEIVED FROM GREEN" if k = 1.
2.0008 To step 2.23.
2.001 Recall item R(1).
2.0011 Discard item R(1) if k = 1.
2.002 To step 2.2 if k ≠ 0.
2.003 Do step 2.005 for n = 1(1)5000.
2.0031 Type "No message from opponent.".
2.0032 Line.
2.004 To step 2.0002.
2.005 Set k = k.
2.2 Page.
2.22 Type "MESSAGE RECEIVED FROM OPPONENT.".
2.23 Type _,_.
2.3 Type part 1 if k ≠ 0.
2.41 Delete part 1.
2.5 Set k = 0.
2.6 File k as item R(1) if X = 0.
2.61 File k as item R(2) if X = 1.
2.62 To step 2.0002 if X = 1.
2.9 Type "END OF MESSAGE.".
2.901 To step 2.001 if X≠0.
2.91 Type _,_,_.
2.911 Type "To REPLY to above message, set K = '1',".
2.912 Type "To continue in MESSAGE Monitoring mode, set K = '0'.".
2.913 Demand K.
2.914 Do part  100 if K = 1.
2.915 To step 2.92 if K = 0.
2.92 Type "Back in message monitoring loop.".
2.99 To step 2.0002.
```

Fig.29—BLUE/RED message monitoring program

```
Type part 100.
100.001 Page.
100.002 Type "MESSAGE GENERATION AND TRANSMISSION PROGRAM.".
100.003 Type _,_,_.
100.1 Set k = 1.
100.2 Type "Preface each line of message with a "Part 1" number.".
100.3 Type "After composing message, type "Go.".".
100.4 Stop.
100.41 Type part 1.
100.42 Type _,_.
100.43 Type "To make CORRECTIONS, set C = '1', else set C = '0'.".
100.44 Demand C.
100.45 To step 100.5 if C = 0.
100.46 Type "Make CORRECTIONS, then type 'Go.'.".
100.47 Stop if C = 1.
100.48 Type "To TRANSMIT message, set K = '1'.".
100.481 Type "To CHECK message again for corrections, set K = '0'.".
100.482 Demand K.
100.483 To step 100.41 if K = 0.
100.484 Line if K = 1.
100.5 Discard item T(1).
100.51 Discard item T(2).
100.6 File k, part 1 as item T(1).
100.601 File k, part 1 as item T(2).
100.61 Type "MESSAGE TRANSMITTED.".
100.7 Delete part 1.
100.8 Type "To shift to MESSAGE MONITORING mode, set K = '1'.".
100.8001 Type "To send ANOTHER MESSAGE, set K = '0'.".
100.801 Demand K.
100.81 Do part 2 if K = 1.
100.82 To step 100.001 if K = 0.
```

Fig. 30—BLUE/RED message generation and transmission program

```
Type part 11.
11.01 Type "JOSS Store-and-Forward COM-SYSTEM Start-up Program for GREEN.".
11.02 Type _,_.
11.03 Type "Give item no. for transmitting messages to BLUE.".
11.031 Demand T(1).
11.032 Type "Give item no. for transmitting messages to RED.".
11.033 Demand T(2).
11.04 Line.
11.05 Type "Give item no. for receiving messages from BLUE to RED.".
11.051 Demand R(1).
11.052 Type "Give item no. for receiving messages from RED to BLUE.".
11.053 Demand R(2).
11.06 Line.
11.07 Type "To begin operation in the RECEIVE mode, set K = '1'.".
11.08 Type "To begin with message to BLUE/RED, set K = '0'.".
11.0801 Demand K.
11.0802 Type "END of GREEN START-UP PROGRAM.".
11.081 To part 200 if K = 0.
11.082 To part 3 if K = 1.
```

Fig.31—GREEN start-up program

```
Type part 3.
3.0001 Type "Entering the MESSAGE MONITORING mode as GREEN.".
3.00011 Do step 3.00013 for n = 1(1)5000.
3.00012 To step 3.0002.
3.00013 Set n = n.
3.0002 Type "Checking for message from BLUE to RED:".
3.0003 Recall item R(1).
3.0004 Type "No message from BLUE to RED." if k = 0.
3.0005 Type _,_ if k = 1.
3.0006 Type "                    Message from  BLUE to RED:" if k = 1.
3.00061 Line.
3.0007 Type part 1 if k = 1.
3.00071 Delete part 1 if k = 1.
3.0008 Type _,_ if k = 1.
3.000801 To step 3.001 if k = 0.
3.00081 Set k = 0.
3.00082 Discard item R(1).
3.00083 File k as item  R(1).
3.001 Type "Checking for message from RED to BLUE:".
3.002 Recall item R(2).
3.003 Type "No message from RED to BLUE." if k = 0.
3.004 Type _,_ if k = 1.
3.0041 Type "                     Message from RED to BLUE:" if k = 1.
3.00411 Line.
3.005 Type part 1 if k = 1.
3.0051 Delete part 1 if k = 1.
3.006 Type _,_ if k = 1.
3.0069 Line.
3.007 To step 3.00011 if k = 0.
3.0071 Set k = 0.
3.0072 Discard item R(2).
3.0073 File k as item R(2).
3.0074 To step 3.00011.
```

Fig.32—GREEN message monitoring program

```
Type part 200.
200.001 Page.
200.002 Type "GREEN MESSAGE GENERATION AND TRANSMISSION PROGRAM.".
200.003 Type _,_,_.
200.1 Set k = 1.
200.2 Type "To send message to BLUE, set S = '1'; to RED, set S = '0'.".
200.21 Demand S.
200.3 Type "Preface each line of message with a "Part 1" number.".
200.31 Type "After composing message, type "Go.".".
200.4 Stop.
200.41 Type part 1.
200.42 Type _,_.
200.43 Type "To make CORRECTIONS, set C = '1', else set C = '0'.".
200.44 Demand C.
200.45 To step 200.5 if C = 0.
200.46 Type "Make corrections, then type "Go.".".
200.47 Stop if C = 1.
200.48 Type "To TRANSMIT message, set K = '1'.".
200.481 Type "To CHECK message again for corrections, set K = '0'.".
200.482 Demand K.
200.483 To step 200.41 if K = 0.
200.484 Line if K = 0.
200.5 Discard item T(1) if S = 1.
200.51 Discard item T(2) if S = 0.
200.6 File k, part 1 as item T(1) if S = 1.
200.61 File k, part 1 as item T(2) if S = 0.
200.62 Type "MESSAGE TRANSMITTED TO BLUE." if S = 1.
200.63 Type "MESSAGE TRANSMITTED TO RED." if S = 0.
200.7 Delete part 1.
200.8 Type "To shift to GREEN MESSAGE MONITORING mode, set K = '1'.".
200.8001 Type "To send ANOTHER MESSAGE, set K = '0'.".
200.801 Demand K.
200.81 Do part 3 if K = 1.
200.82 To step 200.001 if K = 0.
```

Fig.33—GREEN message generation and transmission program

REFERENCES

1. Dalkey, N. C., Simulation of Military Conflict, The RAND Corporation, P-3400, January 1967.

2. Cripwell, F. J., "A Discussion of Computer Assisted Land Combat Games," Proceedings of the Fifth Symposium on War Gaming, published by HRB-Singer for the East Coast War Games Council, August 1966, pp. 255-268.

3. Northrop, G. M., Digital Communications and EDP for an Advanced Tactical Air Control System: A Preliminary Study, The RAND Corporation, RM-4431-PR, January 1965.

4. Digital Message Entry System (DMES) Feasibility Test, Tactical Air Command, USAF Tactical Air Warfare Center, TAC TR 66-7, May 1966.